Overview of Web Content Adaptation

Jérémy Lardon, Mikaël Ates, Christophe Gravier, Jacques Fayolle

OVERVIEW OF WEB CONTENT ADAPTATION

Jérémy Lardon, Mikaël Ates, Christophe Gravier, Jacques Fayolle
DIOM Laboratory, ISTASE, University Jean Monnet, rue du Dr P. Michelon, Saint-Etienne, France
{jeremy.lardon, mickael.ates, christophe.gravier, jacques.fayolle}@univ-st-etienne.fr

Keywords: HCI, web content adaptation, limited browsing devices.

Abstract: Nowadays Internet contents can be reached from a vast set of different devices. We can cite mobile devices (mobile phones, PDAs, smartphones) and more recently TV sets through browser-embedding Set-Top Boxes (STB). The diverse characteristics that define these devices (input, output, processing power, available bandwidth, ...) force content providers to keep as many versions as the number of targeted devices. In this paper, we present the research projects that try to address the content adaptaption.

1 INTRODUCTION

Numerous devices allow the user to browse the Internet these days. Nevertheless the resources found on the Web are designed for personal computers. Taking into account all their characteristics : screen size, resolution, mouse and keyboard, software installed (flash, video players, ...), it clearly appears that mobile devices as well as browser-embedded Set-Top Boxes offer less capabilities in terms of input, output, processing power, or bandwidth. That's the reason why solutions must be found to unite internet contents with the limitations of such devices, henceforth referred as limited devices.

This paper lists previously imagined solutions to solve the above problem above. As a side note and in order to remain brief and revelant, we have made the choice to focus on the last decade and to cite one publication per research project.

In order to structure our presentation, we rely on the Model-Driven Engineering (Gerber et al., 2002). As figure 1 shows, there are two paths to adapt an user interface (UI) to a new device: direct code to code transformation (or transcoding) or re-engineering which passes through three steps : reverse-engineering, model to model transformation and forward-engineering. The main difference between these two approachs is that the former tries to address the transformation in a higher level of abstraction, while the later takes the web page code as support of the transformation.

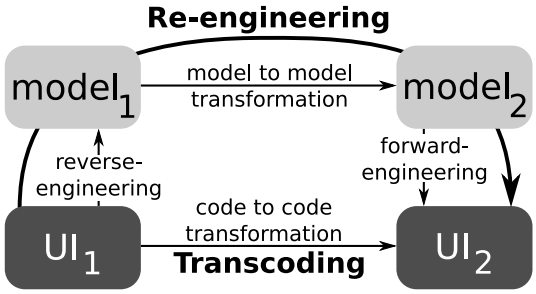

Figure 1: Re-engineering versus transcoding.

Thus our paper is organized as follows: Section 2 presents the publications using the reverse-engineering process. Section 3 deals with the projects in which adaptation techniques are directly used on the code. Finally section 4 concludes.

2 RE-ENGINEERING OF USER INTERFACES

In this section, we summarize what is, to our knowledge, the major research on UI re-engineering in general and the hard point that is reverse-engineering.

2.1 Omini

In (Buttler et al., 2001), the authors present the Omini system that aims to extract objects of interest from web pages. The Omini object extraction process is divided in three phases. The first is the preparation of the web document: retrieval of the page, transformation to a well-formed web document, and convertion to a tag tree representation. Afterwards objects of interest are located in the page. This phase has two steps: the object-rich subtree extraction and the object separator extraction. Finally, objects of interest are extracted thanks to the result of the previous phase.

The main concern of this paper is the identification of object separators. In this goal, five heuristics are compared with their combinaitions. As a consequence, the combinaition of the five heuristics shows the best results on the cached pages from 50 different web sites.

2.2 MORE

(Gaeremynck et al., 2003) focus on discovering the models behind web forms. The main challenge they address is to discover the relationship between strings and widgets. These relations are mutually exclusive as they assume that each entity (string or interactor) plays a unique role, for example a string can't at the same time a caption and a hint for a interactor.

The starting point of the study is a collection of facts extracted from the web page: description of the entities ("S_1 is a string" or "I_2 is an iteractor"), relationships between those entities ("S_1 is a caption for I_2" or "S_1 is a hint for I_2"), ... Facts are then manipulated through a forward chain rule system. Three types of rules are defined : deduction rules to produce new facts from selected facts, exclusion rules to determine if two facts are mutually exclusive and the scoring rules to order facts depending on the properties of the interactors involved.

The model recovery algorithm can be summed up as follows. As long as there remains unprocessed facts, facts are created thanks to the deduction rules. Resulting facts that, when combined, have less chance to create future conflicts (or exclusions) are selected. The second selection amongst them is based on the scoring rules to get the larger set of compatible facts. Only the finally selected facts expand the set. The remainder of the created facts that do not pass the two selections are discarded and the loop continues.

The result is a list of relations between strings and interactors which then can be used to split a form.

2.3 Web RevEnge

Web RevEnge (Paganelli and Paternò, 2003) was developped to automatically extract the task models from a web application, i.e. multiple web pages.

In order to do so, they begin to compute each page. The DOM of the page is parsed to find links, interaction objects (such as <input> tags), their groupings (forms, radio button groups), and finally frames. As the task models are represented in ConcurTasksTrees (Paternò, 2000), task model representations of each page are graphs with a root element and link nodes to other pages.

To build the task model of the whole web application, the process uses the home page as its starting point. All links being represented in the task model, one replaces the internal links (in the same site) with the task models of the targeted pages.

2.4 WARE and WANDA

Even though presented in the same publication (Lucca and Penta, 2005), WARE and WANDA are web application reverse-engineering tools that were developped independently. The former adresses the static analysis of web applications. The latter intervenes upstream by extracting information from the php files.

WARE implements a two-step process. Revelant information is retrieved from the static code (mainly HTML) by extractors. Then abstractors take the previous result as input and abstract them. The final output is a UML representation of the web application.

WANDA does the same work but on dynamic data instead of static data. Dynamic information is collected during web application executions and becomes the support of the extraction that creates UML diagrams.

Bringing them together permits us to identify groups of equivalent dynamically built pages if there are enough execution runs.

2.5 ReversiXML and TransformiXML

ReversiXML and TransformiXML (Bouillon et al., 2005) are respectively a tool to reverse-engineer web pages and a tool to transform abstract representations from one context of use to another.

For this purpose, Bouillon *and al.* takes Cameleon framework (Calvary et al., 2003) as reference for the development process. In order to express any abstraction level of the UI, they rely on UsiXML (http://www.usixml.org).

About the reverse-engineering part, the derivation from code source to any abstraction level is done

thanks to derivation rules, functions interpreted at design- and run-time. The output of this first stage is an UsiXML file that represents the graph of the UI in the selected abstraction level.

The transformation takes place at any level of abstraction. As UsiXML has an underlying graph structure, the model transformation system is equivalent to a graph transformation system based on the theory of graph grammars.

3 TRANSCODING

In constrast to the re-engineering, transcoding directly manipulates the code of the UI. So we put them on the same level and ordered them chronologically.

3.1 Top Gun Wingman

This research project (Fox et al., 1998) was motivated by the use of 3Com PalmPilot as a web browsing device. However the browser used on the PalmPilot is a split web browser, i.e. that it needs a dedicated server to run, in addition to the software on the device. The server side, that operates as a proxy lets workers do the adaptation, which is splited into 4 processes:

- image processing
- HTML processing
- aggregation to build contents from one or more sites on a topic
- zip processor to list the archive contents in HTML

3.2 Digestor

Digestor (Bickmore et al., 1999) aims at filtering and automatically re-authoring web pages to display on small screen devices. We focus on the re-authoring use of this project. The implementation of Digestor takes place between the client and the server, in a proxy server. To perform the transformations on pages, Bickmore and Schilit rely on fifteen techniques grouped in three groups:

1. **Outlining**: displays only section headers as links that point to new pages displaying the texts under the headers,
2. **First sentence elision**: uses the same technique but the first sentences of each block become links to the rest of the block,
3. **Indexed segmentation**: segments the page into sub-pages containing a given number of items,
4. **Table transformation**: splits a page between regions, such as sidebars, headers and footers,
5. **Image reduction or elision**: reduces or suppresses images from the page, replaces images with a reduced image or the ALT text pointing to the original image.

Given these techniques, the question of which techniques to use and in what order remains. The response to this question is a heuristic planner which explores all possibilities and gives them a score estimated from the screen area required to display the new page. The process is recursive until the document version is judged good enough or there is no candidate. In this case the best estimated version is returned.

3.3 Power Browser

(Buyukkokten et al., 2000) also take advantage of a proxy architecture. In addition to transforming the pages, Power Browser manages the navigation. That's why it is more drastic than the previous proxy-based solution about the re-authoring.

Indeed all images are replaced by their ALT property value. In the same way all white spaces are collapsed to save screen space. Tables and lists are reformatted in text block.

In the other hand the navigation is made easier by the use of shortcuts and of tree control to display links.

3.4 Web page structure detection

The approach exposed in (Chen et al., 2003) proposes to facilitate navigation and reading on small screen devices. To do so, a thumbnail is available when the user first requests a page. In the tumbnail, each semantic block is colored and makes possible the choice of what part of the web page the user wants to see in detail.

This approach allows the page to be split into blocks. The block identification part of the process parses the DOM tree to find:

1. high-level content blocks such as tags, header, footer, and left/right side bars,
2. explicit separators: <hr> tags, rows in a table (<tr>), , any tag with border properties,
3. implicit separators by using pattern recognition and clustering on tag names and properties.

4 CONCLUSION

As we saw in this paper, web UI adaptation comes in two flavours: transcoding and re-engineering.

While the former can be considered as more basic, by acting directly on the code, transcoding techniques are easier to implement.

On the contrary, the later relies on higher levels of abstraction. A practical case, in which re-engineering is more suitable, is the adaptation of a page with a <select> list (without a *multiple* attribut) to a device, on which only the <input type="radio"> tag is supported. Both tags denote a list with one selectable option, but staying at a concrete level doesn't permit to know that the underlying interaction object is the same for both tags. Nevertheless the main drawback of the re-engineering is its cost.

Therefore both approaches are complementary: fast low level modifications through transcoding and costly more abstract transformations thanks to re-engineering.

What result would the combination of both approaches give? We plan to explore the possibility to bring them together in our future work in order to answer this question.

REFERENCES

Bickmore, T. W., Girgensohn, A., and Sullivan, J. W. (1999). Web page filtering and re-authoring for mobile users. *Computer Journal*, 42(6):534–46.

Bouillon, L., Limbourg, Q., Vanderdonckt, J., and Michotte, B. (2005). Reverse engineering of web pages based on derivations and transformations. In *Web Congress, 2005. LA-WEB 2005. Third Latin American*.

Buttler, D., Liu, L., and Pu, C. (2001). A fully automated object extraction system for the world wide web. In *Proceedings of the 2001 International Conference on Distrubuted Computing Systems (ICDCS'01)*, pages 361–370, Phoenix, Arizona.

Buyukkokten, O., Garcia-Molina, H., Paepcke, A., and Winograd, T. (2000). Power browser: efficient web browsing for pdas. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 430–437, New York, NY, USA. ACM.

Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., and Vanderdonckt, J. (2003). A unifying reference framework for multi-target user interfaces. *Interacting With Computers Vol. 15/3*, pages 289–308.

Chen, Y., Ma, W.-Y., and Zhang, H.-J. (2003). Detecting web page structure for adaptive viewing on small form factor devices. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 225–233, New York, NY, USA. ACM.

Fox, A., Goldberg, I., Gribble, S. D., and Lee, D. C. (1998). Experience with top gun wingman: A proxy-based graphical web browser for the 3com palmpilot. In *Proceedings of Middleware '98, Lake District, England, September 1998*.

Gaeremynck, Y., Bergman, L. D., and Lau, T. (2003). More for less: model recovery from visual interfaces for multi-device application design. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 69–76, New York, NY, USA. ACM.

Gerber, A., Lawley, M., Raymond, K., Steel, J., and Wood, A. (2002). Transformation: The missing link of mda. In *ICGT '02: Proceedings of the First International Conference on Graph Transformation*, pages 90–105, London, UK. Springer-Verlag.

Lucca, G. A. D. and Penta, M. D. (2005). Integrating static and dynamic analysis to improve the comprehension of existing web applications. In *WSE '05: Proceedings of the Seventh IEEE International Symposium on Web Site Evolution*, pages 87–94, Washington, DC, USA. IEEE Computer Society.

Paganelli, L. and Paternò, F. (2003). A unifying reference framework for multi-target user interfaces. *International Journal of Software Engineering and Knowledge Engineering, World Scientific Publishing 13(2)*, pages 169–189.

Paternò, F. (2000). Model-based design of interactive applications. *Intelligence*, 11(4):26–38.